Izpit iz predmeta Programiranje 1, 30. avgust 2012

Rešitve **vseh** nalog shranite **v eno samo datoteko s končnico .py** in jo oddajte **nezazipano** prek Učilnice, kot ste oddajali domače naloge. Funkcije naj imajo enaka imena in argumente, kot jih predpisuje naloga. Rezultate naj vrnejo, ne izpišejo. Dovoljena je vsa literatura na poljubnih medijih, ves material, ki je objavljen na Učilnici, vključno z objavljenimi programi; njihova uporaba in predelava se ne šteje za prepisovanje.

Pozorno preberi naloge in ne rešuj le na podlagi podanih primerov! Da rešitev ne bi vsebovala trivialnih napak, jo preverite s testi v ločeni datoteki na Učilnici. Za rešitev lahko dobite določeno število točk, tudi če ne prestane vseh testov (in obratno).

Izpit morate pisati na fakultetnih računalnikih, ne lastnih prenosnikih.

Študenti s predolgimi vratovi in podobnimi hibami bodo morali zapustili izpit, katerega opravljanje se bo štelo kot neuspešno. Korektnost poteka izpita bomo nadzorovali s tehničnimi pripomočki in hujše kršitve prijavili disciplinski komisiji.

Čas pisanja: 90 minut.

Nalogi A in B se ne točkujeta, temveč sta **obvezni** za uspešno opravljen izpit. Kdor ju ne reši pravilno, je s tem že padel. Nalogi morate rešiti tako, da predelate SVOJO rešitev domače naloge. Kdor naloge ni uspešno opravil vzame objavljeno rešitev.

Ostale naloge so vredne enako število točk.

A. Površni slovar

Spremeni rešitev naloge Neobčutljivi slovar tako, da bo sicer *razlikovala* med malimi in velikimi črkami ("Kamela" naj ne bo več isto kot "kamela"), pač pa bo upoštevala le prvi štiri znake besede in zanemarila ostale. Torej, "kamela" naj pomeni isti ključ kot "kamen".

B. Velikosti slik

Spremeni izpis programa tako, da se bo začel z velikostjo slike (širina in višina sta široka po osem znakov, med njima pa je " x ", sledil pa bo naslov filma, dolg 50 znakov in poravnan na desno. Primer je spodaj (prva vrstica s številkami služi le pomoči pri štetju in naj je program ne izpisuje!).

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
    1130 x 529                          ameriski proracun energija.png
     968 x 99                                                  nara.jpg
     968 x 1480                                          box-office.png
```

1. Vse po dvakrat

Napiši funkcijo `vse_po_dvakrat(s)`, ki prejme niz `s` in vrne `True`, če se vsak znak, ki se pojavi v nizu, pojavi natačno dvakrat. Presledke ignoriraj.

Primer

```
>>> vse_po_dvakrat("abc add b c")
True
>>> vse_po_dvakrat("ab+c ad b+ c")  # d se pojavi le enkrat
False
>>> vse_po_dvakrat("aabbaacc")      # a se pojavi štirikrat
False
>>> vse_po_dvakrat("")              # vsak znak, ki se pojavi, se pojavi dvakrat...
True
```

2. Kamelja imena

V Pythonu je prišlo v modo pisanje imen s podčrtaji. Tako imamo namesto spremenljivke `prvaCrka` spremenljivko `prva_crka` in namesto funkcije `popraviImeFunkcije` funkcijo `popravi_ime_funkcije`.

Napiši funkcijo `brez_grb(s)`, ki kot argument prejme ime `s`, ki je morda zapisano tako, da se besede začenjajo z velikimi začetnicami, in vrne ime, v katerem so besede ločene s podčrtaji. Pri tem mora paziti na dve izjemi.

- Če je prva črka imena velika, pusti ime pri miru. Tako bo `KameljiRazred` ostal `KameljiRazred` in ne `Kamelji_razred` (ali `kamelji_razred` ali celo `_kamelji_razred`).
- Poleg tega mora pustiti pri miru tiste velike črke, ki jim sledijo velike črke. Tako mora `pretvoriHTML` postati `pretvori_HTML`; dodamo torej podrčtaj, črka pa ostane velika.

Primer

```
>>> brez_grb("kameljaImena")
"kamelja_imena"
>>> brez_grb("nekaDvogrbaKamela")
"neka_dvogrba_kamela"
>>> brez_grb("KameljaImena")
"KameljaImena"
>>> brez_grb("datotekaHTML")
"datoteka_HTML"
```

3. Seštevke

S števili včasih delamo tole: seštejemo njihove števke, nato seštejemo števke vsote in tako naprej, dokler ne dobimo ene same števke. Vzemimo, recimo, 39. Seštejemo števke in dobimo 3+9=12. Tudi v dvanajst seštejemo vse (torej obe) števki in dobimo 1+2=3. Potem rečemo, da številu 39 pripada števka 3.

Napiši funkcijo `sestevke(a, b)`, ki za vsa števila od `a` do *vključno* `b` izračuna pripadajoče števke ter rezultat vrne v obliki slovarja. Ključi slovarja naj bodo števke, vrednosti pa seznami števil, ki jim pripada ta števka.

Primer

```
>>> sestevke(17, 27)
{8: [17, 26], 9: [18, 27], 1: [19], 2: [20], 3: [21], 4: [22], 5: [23], 6: [24], 7:
[25]}
>>> sestevke(99999, 99999)
{9: [99999]}
```

4. Liki

Dani so naslednji razredi, ki predstavljajo like.

```
class Pravokotnik:
  def __init__(self, a, b):
    self.a, self.b = a, b

class Krog:
  def __init__(self, r):
    self.r = r

class Trikotnik:
  def __init__(self, a, b, c):
    self.a, self.b, self.c = a, b, c
```

Poleg tega je podana funkcija, ki kot argument sprejme seznam likov in kot rezultat vrne njihovo ploščino.

```
def skupna_ploscina(liki):
    p = 0
    for lik in liki:
        p += lik.ploscina()
    return p
```

Gornjih definicij ti ni potrebno pretipkavati: najdeš jih v datoteki s testi.

Dopolni definicije razredov (funkcijo `skupna_ploscina` je prepovedano spreminjati!) tako, da bo funkcija `skupna_ploscina` delovala. (Opomba: ploščino trikotnika izračunaš po Heronovem obrazcu, $p = \sqrt{s(s-a)(s-b)(s-c)}$, kjer je s pol obsega, $s = (a+b+c)/2$.

Primer

```
>>> trikotnik = Trikotnik(2, 4, 5)
>>> pravokotnik = Pravokotnik(4, 5)
>>> krog = Krog(3)
>>> pravokotnik = Pravokotnik(1, 1)
>>> skupna_ploscina([trikotnik, pravokotnik, krog, pravokotnik2)
53.0740049207008
```

5. Črkovalnik

Napiši funkcijo `spell(fname, fdict)`, ki vrne seznam vseh besed v datoteki z imenom `fname`, ki niso v slovarju, shranjenem v datoteki z imenom `fdict`. Vsaka beseda v slovarju je zapisana v svoji vrstici in z malimi črkami. Besede v seznamu, ki ga vrne funkcija, naj bodo izpisane z enakimi črkami kot so v originalnem besedilu (glej besedo "Dans" v primeru).

esej.txt	**slovar.txt**
Dans je lep dan. Jutri bo prav tako sijal sonc.	dan danes je izpit jutri lep prav tako sijalo sonce mesec Jupiter hrošč

```
>>> spell("esej.txt", "slovar.txt")
['Dans', 'bo', 'sijal', 'sonc']
```